15

Regular Approximations through Labeled Bracketing (revised version)

ANSSI YLI-JYRÄ

15.1 Introduction

Regular approximations for context-free languages have a wide use as language models in natural language and speech processing. Since the famous work by Pereira and Wright (1991), several approximation methods have become available (cf. Nederhof 2000). However, there are further approximation methods for *structured languages* whose strings indicate the constituent structure by means of brackets. Such methods may have a wide application area because of the recently renewed interest in structured languages in formal language theory (e.g. Kappes 1998), in XML document processing (e.g. Berstel and Boasson 2000), in finite-state methods in NLP (Roche 1996, Yli-Jyrä 2003a), and in dependency syntax (Yli-Jyrä 2003b).

A good regular approximation for context-free grammars should have at least the following properties: (i) It is well understood. (ii) It admits a compact representation that is so small that it can be stored into realistic computers (a compact representation consists of sub-automata that are combined lazily). (iii) It assigns bracketings with an accuracy that is practically sufficient for replacing the original parsing grammar. (iv) Its representation is easy to inspect and to modify (Nederhof 1997). (v) Parsing with the approximation does not lead to a combinatorial explosion from the compact representation.

The exactness up to any predefined center-embedding depth is an important property when we measure accuracy of approximations that assign bracketings. Schematically, in a phrase $[\alpha\beta\gamma]$, the substring β is a *center-embedding*, if $\alpha \neq \lambda$ and $\gamma \neq \lambda$. The *center-embedding depth* measures how

Submitted to the post-proceedings of FGVienna.
(editors of the proceedings).

many phrases are nested as center-embeddings into each other. A *subset (superset) approximation* (Nederhof 2000) is said to be *exact up to any given center-embedding depth* d if it rejects (accepts) a string w only if w is rejected (accepted) by the original grammar or if the center-embedding depth of w is beyond the bound d.

A paradox in the earlier approximation methods is that they fail to combine a small compact representation with a sufficient exactness. For example, the approximation by Mohri and Nederhof (2001) admits a very compact representation, but it is a very coarse superset approximation for parsing purposes. The approximation by Johnson (1998) is exact up to any given depth of center-embedding, but the size of its compact representation grows very fast compared to the number of rules in the source grammar (Nederhof 2000: Figure 12: the methods LC2,LC3,LC4). Nevertheless, it would be interesting to find regular approximations that combine a small compact representation with a sufficient bound for exactness. Such grammars would be sufficient for context-free parsing under a reasonable performance restriction for the center-embedding.

This paper presents an approximation method that is based on a new *representation theorem* for context-free languages. According to it, any context-free language can be represented as a homomorphic image of an intersection of a set of *constraint languages* defining properties of valid labeled bracketings. The intersected languages of the new theorem differ from the ones used in the famous theorem by Chomsky and Schützenberger (1963). If these constraint languages are restricted to make them regular, we obtain a new kind of compact representation for regular approximations. The resulting approximation can be chosen to be either a subset or a superset of the original context-free language.

The subset and superset approximations obtained in this paper solve the paradox between the practical size of the representation and the sufficient bound for exactness. The approximations have properties (i) – (iv). As to property (v), we do not know yet how the current approximation behaves. The compact representations for the approximations constitute a subclass of *finite-state intersection grammars (FSIG)*. In general, FSIGs have a danger of combinatorial explosion during parsing (Tapanainen 1997), but it is tantalizing to see whether the currently presented subclass admits parsing algorithms that eliminate the problem.

The paper is structured as follows. In Section 15.2, we will introduce *context-free bracketing grammars (CFBGs)*. Section 15.3 discusses a constraint-based approach to language specification. This approach gives rise to *flat CFBGs*, which are introduced in Section 15.4. Section 15.5 contains a representation theorem that connects flat CFBGs with CFBGs. In section 15.6, we will explain how various regular approximations can be obtained from a flat

CFBG. We conclude in Section 15.7.

15.2 Generating Labeled Bracketings

15.2.1 Extended Context-Free Grammars

We have chosen extended context-free grammars (ECFGs) as the starting point, because this leads to wider applicability of the results. ECFGs were introduced by Thatcher (1967). It is well known that they generate exactly the context-free languages. Furthermore, it could be shown that for any standard context-free grammar there exists a strongly equivalent ECFG. Formally, ECFGs are defined as follows:

Definition 1 If E is a *regular expression* (Salomaa 1973), its *language* is denoted with $L(E)$. The empty string is λ. When L is a language, expression $L?$ denotes the language $L \cup \{\lambda\}$.

Definition 2 *Extended context-free grammar* is a quadruple $G = (V_N, V_T, P, S)$, where V_N is called *nonterminal alphabet* and V_T *terminal alphabet*, $S \in V_N$ is called the *start symbol* and P is a finite set of *production schemas* — one for each nonterminal; the alphabets V_N and V_T are finite and disjoint and their union $V = V_N \cup V_T$ is called the *total alphabet*; each production schema is of the form $X \rightarrow E_X$, where X is in V_N and E_X is a regular expression over the alphabet V.

Let w and w' be strings over V. We say that $w \overset{\Rightarrow}{G} w'$ or w' is *directly derivable from* w in grammar G if there are strings $u, v, x \in V^*$, a nonterminal $X \in V_N$ and a production schema $X \rightarrow E_X \in P$ such that $w = uXv$, $w' = uxv$ and $x \in L(E_X)$. The *language* $L(G)$ *generated by* G is $\{w \mid w \in V_T^*, S \overset{\Rightarrow}{G}{}^+ w\}$, where $\overset{\Rightarrow}{G}{}^+$ denotes the transitive closure of the relation $\overset{\Rightarrow}{G}$.

15.2.2 Context-Free Bracketing Grammars (CFBGs)

We want to encode constituent structures by means of labeled bracketing. For standard CFGs, this could be accomplished with CFGs that generate the *structured context-free languages* of Ritchie and Springsteel (1972). However, we define *context-free bracketing grammars* because this new class of grammars gives us more flexibility.

Definition 3 A *context-free bracketing grammar (CFBG)* is an ECFG $G = (V_N, V_T, P, S)$, where

- V_N contains three disjoint subsets:
  - N is the set of *basic nonterminals*,
  - $\overleftarrow{N} = \{\overleftarrow{X} | X \in N\}$ is the set of *left-branching nonterminals*,
  - $\overrightarrow{N} = \{\overrightarrow{X} | X \in N\}$ is the set of *right-branching nonterminals*,
- V_T is the set of terminal symbols containing five disjoint subsets:

- $B_L = \{\, [_X \mid X \in N\}$ and $B_R = \{\,]_X \mid X \in N\}$ are, respectively, the set of *left square brackets* and the set of *right square brackets*,
- $B_l = \{\, \langle_X \mid X \in N\}$ and $B_r = \{\, \rangle_X \mid X \in N\}$ are, respectively, the set of *left angle brackets* and the set of *right angle brackets*,
- $\Sigma = V_T - (B_L \cup B_R \cup B_l \cup B_r)$

- P contains three disjoint sets of production schemata:

$$\{\overleftarrow{X} \rightarrow \quad E_{\overleftarrow{X}}\rangle_X \mid \overleftarrow{X} \in \overleftarrow{N} \text{ and } L(E_{\overleftarrow{X}}) \subseteq (\overleftarrow{N}?)(\Sigma \cup N)^* \quad\} \cup$$
$$\{X \rightarrow [_X\, E_X\,]_X \mid X \in N \text{ and } L(E_X\,) \subseteq (\overleftarrow{N}?)(\Sigma \cup N)^*(\overrightarrow{N}?)\} \cup$$
$$\{\overrightarrow{X} \rightarrow \langle_X E_{\overrightarrow{X}} \quad \mid \overrightarrow{X} \in \overrightarrow{N} \text{ and } L(E_{\overrightarrow{X}}) \subseteq \quad (\Sigma \cup N)^*(\overrightarrow{N}?)\}.$$

CFBGs generate *context-free bracketing languages (CFBLs)*, which are properly included into the context-free languages.

Definition 4 Let $G_0 = (N, \Sigma, P_0, S)$ be an ECFG. Without loss of genericity we can assume that for every $X \in N$ there are new symbols $\overleftarrow{X}$, $\overrightarrow{X}$, $[_X$, $]_X$, $\langle_X$, and $\rangle_X$ such that they do not belong to $N \cup \Sigma$.

The *canonically obtained CFBG* $G = (V_N, V_T, P, S)$ is constructed as follows: The set of basic nonterminals N fixes, by Definition 3, $\overleftarrow{N}$, $\overrightarrow{N}$, B_L, B_R, B_l and B_r. Now $V_N = N \cup \overleftarrow{N} \cup \overrightarrow{N}$ and $V_T = \Sigma \cup B_L \cup B_R \cup B_l \cup B_r$. Let the set of production schemas P be $\{X \rightarrow [_X E_X]_X \mid X \rightarrow E_X \in P_0\} \cup \{X \rightarrow \langle_X \emptyset \rangle_X \mid X \in \overleftarrow{N}\} \cup \{X \rightarrow \langle_X \emptyset \mid X \in \overrightarrow{N}\}$.

Let h_1 be the homomorphism $h_1 : V_T^* \rightarrow \Sigma^*$ such that $h_1(a) = a$ for every $a \in \Sigma$, and $h_1(V_T - \Sigma) = \lambda$.

Theorem 1 *Every context-free language L' is a homomorphic image of a CFBL generated by a canonically obtained CFBG.*

Proof. Let G_0 be an ECFG generating L', and G the CFBG canonically obtained from grammar G_0. Obviously, $L' = h_1(L(G))$. □

Recognition of bracketed strings with a CFBG G is easy: the process reduces to validation of the constituent structure indicated by the bracketing. However, in many practical settings, the strings are not bracketed in advance. All the labeled bracketings for an unbracketed string w are obtained by computing $L(G) \cap h_1^{-1}(w)$. The resulting language is context-free, because $h_1^{-1}(w)$ is a regular language and the context-free languages are closed under intersection with regular languages (Harrison 1978).

Historical note 1 Other kinds of grammars that generate structured languages have been defined in the literature. So-called *bracketed context-free grammars* (Ginsburg and Harrison 1967) differ from CFBGs in two ways: in them, (i) the rules are productions rather than production schemata (i.e. the right-hand side ω of each grammar rule $X \rightarrow \omega$ is a string rather than a regular expression), (ii) there may be several productions for each nonterminal;

they are identified with different brackets. Other structured grammars include the *Chomsky-Schützenberger grammars* (cf. Berstel and Boasson 2000), *very simple grammars* (Korenjak and Hopcroft 1966), *parenthesis or parenthesized grammars* (McNaughton 1967, Salomaa 1973), *multi-parenthesis grammars* (Ginsburg et al. 1975), *generalized parenthesis grammars* (Takahashi 1975), *bracketing transduction grammars* (Wu 1995), *bracketed contextual grammars* (Kappes 1998) and *XML grammars* (Berstel and Boasson 2000).

15.2.3 CFBGs that Generate Reduced Bracketing

In canonically obtained CFBGs, the brackets occurring on both sides of each phrase force all the embedded phrases to constitute a center-embedding. These extra center-embeddings involve a disadvantage from the practical point of view, especially when the language is approximated through pushdown automata whose storage size is restricted.

Definition 5 In *full bracketing*, no bracket is shared by nested constituents. In *reduced bracketing*, a bracket symbol is never repeated, and two kinds of brackets are used; square brackets are balanced while the angle brackets aren't. The following two lines illustrate the two formats for bracketed strings:

$$\begin{array}{ll} [[[[[a]b]c]d][e][[f]] & g[h[i[[j[k[l[m]]]]n]]]] \\ [\quad a\rangle b\rangle c\rangle d\rangle[e][\langle f\ ] & g\langle h\langle i\langle[j\langle k\langle l\langle m\quad]n\quad] \end{array}$$

A repeated square bracket on the upper line corresponds to only a single square bracket on the lower line. To maintain the balanced square bracketing on the lower line, some of the remaining brackets have been changed to angle brackets. The important thing is that the format of the upper line is easily *restorable* from the lower line (using a deterministic linear-time algorithm). So, we can say that the two bracketings denote the same constituent structure.

Definition 6 Define rational transductions $s, t : V^* \to V^*$ as follows:

$$s(w) = \begin{cases} \overleftarrow{X}v, & \text{if } w = Xv \text{ for some } X \in N, v \in V^*; \\ w, & \text{otherwise.} \end{cases}$$

$$t(w) = \begin{cases} v\overrightarrow{X}, & \text{if } w = vX \text{ for some } X \in N, v \in V^*; \\ w, & \text{otherwise.} \end{cases}$$

Theorem 2 *For every canonically obtained CFBG $G = (V_N, V_T, P, S)$ there exists a CFBG $G' = (V_N, V_T, P', S)$ that uses reduced bracketing so that $L(G)$ is restorable from $L(G')$.*

Proof. For every production schema $X \to [_X E_X]_X \in P$, let $E'_{\overleftarrow{X}}$, E'_X and $E'_{\overrightarrow{X}}$ be regular expressions denoting, respectively, the languages $s(L(E_{\overleftarrow{X}}))$, $s(t(L(E_X)))$ and $t(L(E_{\overrightarrow{X}}))$. The set of production schemata P' is con-

structed as follows:

$$\begin{aligned} P' = \{\ \overleftarrow{X} \rightarrow \quad & E'_{\overleftarrow{X}}\rangle_X \mid X \in N\} \cup \\ \{\ X \ \rightarrow & [XE'_X]_X \mid X \in N\} \cup \\ \{\ \overrightarrow{X} \rightarrow & \langle_X E'_{\overrightarrow{X}} \quad \mid X \in N\} \end{aligned}$$

It is easy to see that the phrases that immediately contain square brackets disallow repeating square brackets in those immediate constituents that are in the outermost positions inside the local brackets; in the outermost positions the immediate constituents switch to "widow" angle brackets. Thus, the resulting bracketing is reduced.

Finally, it remains to be shown that the full bracketing is easily restorable in each string. For each angle bracket, the respective pair should be inserted next to the square bracket that embraces the angle-bracketed constituent. It is easy to see that this can be done for each bracketed string by means of a deterministic linear-time algorithm. □

Historical note 2 The historical roots of the reduced bracketing goes back to the ideas of Krauwer and des Tombe (1981) or even further (cf. Chomsky 1963: Section 4). According to Nederhof (2000), similar ideas have been put forward by Langendoen and Langsam (1987), Pullman (1986), Black (1989), Johnson (1996). The method by Johnson (1996) applies only to binary context-free productions, while our method is more general and applies to production schemas.

15.3 A Language Representation via Intersection

Our goal is to eliminate rewriting that is based on rewriting rules and to define a new representation for CFBLs using the intersection operation. One technique for representing context-free languages by means of intersection is given by the Chomsky-Schützenberger Theorem.

Definition 7 The *Dyck language* (i.e. *semi-Dyck language* according to Harrison (1978)) D_n over the alphabet $L_n \cup R_n$, where $L_n = \{[_1, [_2, \ldots, [_n\}$ and $R_n = \{]_1,]_2, \ldots,]_n\}$, is the language generated by the context-free grammar $S \rightarrow \lambda \mid SS \mid [_1 S]_1 \mid [_2 S]_2 \mid \ldots \mid [_n S]_n$.

Theorem 3 (Chomsky and Schützenberger 1963) *Every context-free language L is a homomorphic image of the intersection of a Dyck language D and a regular language R.*

If a grammar G is in the Chomsky normal form, it is easy to construct an equivalent representation $h_2(D \cap R)$. The homomorphism h_2 substitutes brackets either with λ or with the letters of L. The strings of $D \cap R$ are bracketed so that each bracket indicates either a terminal symbol or a production. Intuitively, the regular language R takes care of local properties. The Dyck

language D, in turn, takes care of non-local properties. A detailed proof can be found e.g. in Salomaa (1973).

The Chomsky-Schützenberger representation eliminates the need for language specific rewriting rules. It describes all the properties of the context-free language using constraints on strings. However, in order to implement parsing of CFBG in this way, we have to use two morphisms, h_1 and h_2, which results in an unnecessarily complex formula: $h_2(D \cap R \cap h_2^{-1}(h_1^{-1}(w)))$

Definition 8 Given an alphabet V and languages $C, Lc, Rc \subseteq V^*$, the *context restriction constraint* $C \Rightarrow Lc \_ Rc$ (Koskenniemi 1983) denotes the language $V^* - ((V^* - V^*Lc)CV^* \cup V^*C(V^* - RcV^*))$.

Let the homomorphism $h_3 : (L_n \cup R_n)^* \rightarrow \{[_1,]_1\}^*$ be defined in such a way that $h_3(L_n) = \{[_1\}$ and $h_3(R_n) = \{]_1\}$. Denote the inverse homomorphic image $h_3^{-1}(D_1)$ with $\widehat{D}$.

Theorem 4 (Wrathall 1977) *Any Dyck language D_n equals to an intersection of $\widehat{D}$ and n context restriction constraints $]_i \Rightarrow [_i \widehat{D} \_ \lambda$, $1 \leq i \leq n$.*

Wrathall's theorem demonstrates that bracket labels can be matched using distributed constraints instead of D_n. In the following section, we use other kinds of distributed constraints to achieve the same effect.

15.4 Flat CF Bracketing Grammars (FCFBGs)

The Chomsky-Schützenberger theorem involves a hidden structured language. There is, however, a direct way to represent CFBL by means of constraint languages.

Definition 9 Let the homomorphism $h_4 : V_T^* \rightarrow \{[_1,]_1\}^*$ be defined in such a way that $h_4(B_L) = \{[_1\}$, $h_4(B_R) = \{]_1\}$, and $h_4(\Sigma \cup B_l \cup B_r) = \{\lambda\}$. The language $h_4^{-1}(D_1)$ is denoted with D'. Substitution f_1 is a mapping from $(V \cup \{\Delta\})^*$ to subsets of V^* defined so that it replaces the special symbol Δ with the language D'.

Given an alphabet V and regular expressions Lc, Rc, C over $V \cup \{\Delta\}$, the *bracketing restriction constraint* $\#Lc \_ Rc\# \Rightarrow C$ denotes the language $L(\#Lc \_ Rc\# \Rightarrow C) = \{v \mid v \in D' \wedge \forall x(x \in D' \wedge v \in f_1(L(Lc)\, x\, L(Rc)) \Longrightarrow x \in f_1(L(C)))\}$. When $L(Lc) = V^*Lb$ and $L(Rc) = RbV^*$, such that $Lb \subseteq (B_L \cup B_l)$ and $Rb \subseteq (B_R \cup B_r)$, the constraint $\#Lc \_ Rc\# \Rightarrow C$ can be written more conveniently as $Lb \_ Rb \Rightarrow C$.

A flat CFBG is a triple $G = (N, V_T, K)$, where N is a finite set of *bracket labels*, V_T is a finite alphabet, and K is a finite set of *bracketing restriction constraints*. Alphabet V_T contains five disjoint subsets: $B_L = \{[_X | X \in N\}$, $B_R = \{]_X | X \in N\}$, $B_l = \{\langle_X | X \in N\}$, $B_r = \{\rangle_X | X \in N\}$, and $\Sigma = V_T - (B_L \cup B_R \cup B_l \cup B_r)$. The language $L(G) \subseteq V_T^*$ described

by G is $\cap_{K_i \in K} L(K_i)$.

In the following, we assume that alphabets V_N, V_T, N, $\overleftarrow{N}$, $\overrightarrow{N}$, B_L, B_R, B_l, B_r, Σ are defined by the context.

Definition 10 Substitution f_2 is a mapping from V to $(V \cup \{\Delta\})^*$ defined as follows:

$$f_2(X) = \begin{cases} \Delta\rangle_X & \text{if } X \in \overleftarrow{N}; \\ [_X\, \Delta]_X & \text{if } X \in N; \\ \langle_X \Delta & \text{if } X \in \overrightarrow{N}; \\ X & \text{otherwise.} \end{cases}$$

The mapping f_2 is extended to regular expressions as follows: $f_2(\lambda) = \lambda$; $f_2(\emptyset) = \emptyset$; $f_2(xy) = f_2(x)f_2(y)$; $f_2(x^*) = f_2(x)^*$; $f_2(x|y) = f_2(x)|f_2(y)$.

A *canonically obtained flat CFBG* $G' = (N, V_T, K)$ is associated with every CFBG $G = (V_N, V_T, P, S)$ as follows: N is the basic nonterminal alphabet of G. Let

$$\begin{aligned} K = \{ \;\; \#\_\# \;\; &\Rightarrow f_2(S) \;\} \qquad \cup \\ \{ \;\; [_X\, \_\,]_X \;\; &\Rightarrow f_2(E_X) \;\mid X \in N, X \to [_X E_X]_X \;\in P \;\} \cup \\ \{ \; B_L\, \_\, \rangle_X \;\; &\Rightarrow f_2(E_{\overleftarrow{X}}) \mid X \in N, \overleftarrow{X} \to \;\; E_{\overleftarrow{X}}\rangle_X \;\in P \;\} \cup \\ \{ \;\; \langle_X \_\, B_R \Rightarrow f_2(E_{\overrightarrow{X}}) &\mid X \in N, \overrightarrow{X} \to \langle_X E_{\overrightarrow{X}} \;\; \in P \;\} \end{aligned}$$

Example. Let G_1 be a CFBG with the start symbol S and the following set of production schemata:

$$\begin{array}{lcl} S & \to & [_S\, NP \;\; VP\,]_S \\ VP & \to & [_{VP}\, V\,]_{VP} \mid [_{VP}\, V \;\; NP\,]_{VP} \\ NP & \to & [_{NP}\, \text{Jim}\,]_{NP} \mid [_{NP}\, \text{Sue}\,]_{NP} \\ V & \to & [_V\, \text{ran}\,]_V \end{array}$$

These production schemata and the start symbol S give rise to the following set of bracketing restriction constraints:

$$\begin{array}{lcl} \#\_\# & \Rightarrow & [_S \Delta]_S \\ [_S\, \_\,]_S & \Rightarrow & [_{NP} \Delta]_{NP} [_{VP} \Delta]_{VP} \\ [_{VP}\, \_\,]_{VP} & \Rightarrow & [_V \Delta]_V \;\mid\; [_V \Delta]_V [_{NP} \Delta]_{NP} \\ [_{NP}\, \_\,]_{NP} & \Rightarrow & \text{Jim} \mid \text{Sue} \\ [_V\, \_\,]_V & \Rightarrow & \text{ran} \end{array}$$

15.5 Representing CFLs with FCFBGs

Lemma 5 *The languages D' and $L(\#\;\_\;\#\;\Rightarrow\; f_2(S))$ are generated by ECFGs. Furthermore, for each bracket label $X \in N$, the language $L([_X\, \_\,]_X \Rightarrow f_2(E_X))$ is generated by an ECFG.*

Proof. Let $E_{\neq} = \cup_{X,Y \in N, X \neq Y} [_X \Delta\;]_Y$. D' is generated by an ECFG $G_{D'} = (\{\Delta\}, V_T, P_{D'}, \Delta)$, where $P_{D'} = \{\Delta \to \lambda | \Delta\Delta | \Sigma | B_l | B_r | B_L \Delta B_R\}$.

$L(\#\ \_\ \# \Rightarrow f_2(S))$ is generated by an ECFG $G_\# = (\{S, \Delta\}, V_T, P_\#, S)$, where $P_\# = \{\ S \rightarrow [_S \Delta]_S,\ \Delta \rightarrow \lambda|\Delta\Delta|\Sigma|B_l|B_r|B_L\Delta B_R\ \}$. $L([_X\ \_\ ]_X \Rightarrow f_2(E_X))$ is generated by an ECFG $G_X = (N \cup \{\Delta\}, V_T, P_X, \Delta)$, where $P_X = \{\Delta \rightarrow \lambda|\Delta\Delta|\Sigma|B_l|B_r|E_{\neq}|N\} \cup \{X \rightarrow [_X E_X]_X\} \cup \{Y \rightarrow [_Y \Delta]_Y \mid Y \in N, Y \neq X\}$. □

Lemma 6 *Let $G' = (N, V_T, K)$ a flat CFBG canonically obtained from a CFBG $G = (V_N, V_T, P, S)$ that is canonically obtained from an ECFG. $L(G') = L(G)$.*

Proof. We first show that the language $L(G')$ is generated by an ECFG. By Lemma 5, each constraint $L([_X\ \_\ ]_X \Rightarrow f_2(E_X))$ is generated by an ECFG G_X. The intersection $\cap_{X \in N} L(G_X)$ is generated by an ECFG $G_N = (N \cup \{\Delta\}, V_T, P_N, \Delta)$, where $P_N = \{\Delta \rightarrow \lambda|\Delta\Delta|\Sigma|B_l|B_r|E_{\neq}|N\} \cup \{X \rightarrow [_X E_X]_X \mid X \in N\}$. The intersection $L(G')$ of $L(G_N)$ and $L(G_\#)$ is generated by a CFBG $G'' = (V_N, V_T, P', S)$, where $P' = \{X \rightarrow [_X E_X]_X \mid X \in N\} \cup \{X \rightarrow \emptyset\rangle_X \mid X \in \overleftarrow{N}\} \cup \{X \rightarrow \langle_X \emptyset \mid X \in \overrightarrow{N}\}$. Because $G'' = G$, it holds that $L(G') = L(G)$. □

Theorem 7 *Every context-free language L' is a homomorphic image of an intersection of bracketing restriction constraints.*

Proof. Let G_0 be an ECFG generating L' and G the CFBG canonically obtained from G_0. By Theorem 1, L' is a homomorphic image of a $L(G)$. A flat CFBG G' canonically obtained from G is equivalent to G by Theorem 6. So, the language $L(G)$ equals to an intersection of bracketing restriction constraints. □

15.6 Obtaining Approximations from FCFBGs

FCFBGs have the advantage that they allow us to define useful regular approximations simply by altering the language denoted by Δ: if a regular language is substituted for Δ, then the whole grammar becomes regular.

We define approximations for D' by restricting the number of nested brackets. The series of subset approximations $A_0, A_1, A_2, \ldots$ is defined inductively as follows: $A_0 = (\Sigma \cup B_l \cup B_r)^*$, $A_i = A_{i-1}(B_L A_{i-1} B_R\, A_{i-1})^*$, for $i = 1, 2, \ldots$. A schema of the finite automaton that accepts A_k is shown in Figure 1a. In NLP, a plausible setting for k is probably near to 5 (Johnson 1998). It is also possible to define a series of superset approximations $A'_0, A'_1, A'_2, \ldots$; the schema of the automaton that accepts A'_k is shown in Figure 1b.

It is possible to analyze the descriptive complexity of the resulting approximations. *Generalized regular expressions* are regular expressions with the complement operation. Languages like $A_0, A_1, A_2, \ldots$ are captured with

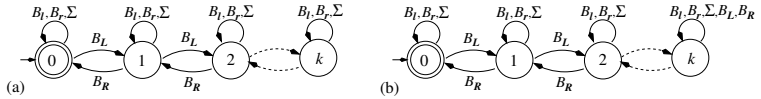

FIGURE 1 A schematic representation of automata for A_k and A'_k.

generalized regular expressions without Kleene's star (Brzozowski and Knast 1978, Yli-Jyrä 2003a). In addition to this, the bracketing restriction constraints can be written with star-free generalized regular expressions. Thus, the whole approximation reduces to a star-free regular language. Our conjecture is that the *dot-depth* of this language is decidable, although, in general, it is an open problem to decide whether a regular language has the dot-depth n (Pin 2003).

The bracketing restriction constraints of a FCFBG correspond to generalized regular expressions in the approximation. Let $\widetilde{D}$ be the approximated language used for D'. Substitution f_3 is a mapping from $(V_T \cup \{\Delta\})^*$ to subsets of V_T^* defined so that it replaces the special symbol Δ with the language $\widetilde{D}$. Mapping f_3 is extended to regular expressions in the same manner as f_2 (Page 8). The subset approximation of the bracketing restriction constraint $\#Lc \_ Rc\# \Rightarrow f_2(E)$ can be compiled using the following formula:

$$\widetilde{D} - Lc(\widetilde{D} - f_3(f_2(E)))Rc$$

This compilation formula can be varied in many ways. When $\widetilde{D} = A_k$, the constraint $Lb \_ Rb \Rightarrow f_2(E)$, where $L(Lb) \subseteq B_L$, and $L(Rb) \subseteq B_R$, is equivalent to $A_k - V_T^* Lc(A_{k-1} - f_3(f_2(E)))RcV_T^*$. Furthermore, if we intersect the language of each bracketing restriction constraint with context restriction constraints $Lb \Rightarrow \lambda \_ A_k Rb$ and $Rb \Rightarrow LbA_k \_ \lambda$ over the alphabet V_T, we get smaller constraint languages and smaller finite automata. This does not affect the language of the whole grammar. In addition to this, equivalent but substantially more efficient ways to compile bracketing restriction constraints exist but they are not reported here.

For any regular language L', its *state complexity* is the size of the minimal deterministic finite automaton that recognizes L'. On the basis of our initial experiments, we have reached to the following generalization for the state complexity of bracketing restriction constraints: If a bracketing restriction constraint is of the form $[_X \_]_X \Rightarrow f_2(E)$, $L(E) \subseteq N^*$, m is the state complexity of $L(f_2(E))$ and t is the number of transitions in the minimal automaton accepting $L(E)$, then the state complexity $s(k)$ of the approximated constraint language

$$L(A_k - V_T^* \, [_X \, (A_k - f_3(f_2(E))) \,]_X \, V_T^*) \cap$$
$$(\, [_X \Rightarrow \lambda \_ A_k]_X \,) \cap (\,]_X \Rightarrow [_X A_k \_ \lambda \,)$$

is estimated inductively as follows: $s(0) = 1$, $s(1) = 2$, and $s(k)$ in $O(s(k -$

$1) + ts(k-2) + m)$. For example, if $E = \{YZ\}$ and $Y, Z \neq X$, we obtain $m = 7$, $t = 2$, $s(2) = 8$, $s(3) = 16$, $s(4) = 36$, $s(5) = 72$ etc. We can also obtain a state complexity that is quadratic in k, if we split each constraint into k sub-constraints so that each sub-constraint checks one nesting level only; each sub-constraint will have a linear state complexity according to k.

Certain computationally cheap changes in the formula improve the accuracy of the resulting constraint languages. If we use the formula $V_T^* - Lc(A_k - f_3(f_2(E)))Rc$, the constraint languages become larger, but this does not affect the language of the whole grammar. The accuracy of the whole grammar can be improved if the constraint languages are compiled so that they have different depth bounds for nested self-embeddings and nested center-embeddings. A *self-embedding* is a center-embedding, where a phrase is center-embedded into a phrase of the same category.

The standard compact representations are based on lazy substitution of transitions with sub-automata (Mohri and Pereira 1998, Nederhof 2000). Their primary purpose is to improve efficiency of off-line construction of finite automata. However, the compact representation may also be used on-line for processing of input (Mohri and Nederhof 2001). Our compact representation is used on-line, but it is different: the sub-automata accepting the individual constraint languages are *intersected* rather than substituted. Parsing in our representation is based on satisfiability of all the finite-state constraints at the same time. The approach is open to extensions where the domain of constraints is not necessarily restricted to local phrase structures.

When the approximation is derived from a FCFBG that uses reduced bracketing, we gain an essential advantage over regular approximations that assign full bracketing to the strings. For example, a small-scale regular approximation (available upon request) analyzed PP-attachment more efficiently and accurately using reduced bracketing.

15.7 Conclusion

We have suggested a new constraint-based representation for context-free sets of bracketed strings. The new representation lends itself for a direct construction of some regular approximations.

The obtained regular approximations have a number of favorable properties: (i) they admit a small compact representation, (ii) they handle tail-recursion appropriately and they are exact up to a predefined depth of center-embedding, and (iii) they can be represented by means of simple regular expressions.

The compact representation has been implemented and tested successfully in practice. Parsing with full-scale grammars *may* require special intersection algorithms to avoid combinatorial explosion.

Acknowledgments

The author is indebted to the anonymous referees and the FGVienna audience for comments and questions that helped to improve the quality of this paper. The research has been funded by the Nordic Academy of Advanced Study (NorFA).

References

Berstel, Jean and Luc Boasson. 2000. XML grammars. In *MFCS 2000, LNCS 1983*, pages 182–191. Springer.

Black, Alan W. 1989. Finite state machines from feature grammars. In M. Tomita, ed., *International Workshop on Parsing Technologies*, pages 277–285. Pittsburgh, Pennsylvania: Carnegie Mellon Univeristy Press.

Brzozowski, Janusz A. and Robert Knast. 1978. The dot-depth hierarchy of star-free languages is infinite.

Chomsky, Noam. 1963. Formal properties of grammars. In R. Luce, R. R. Bush, and E. Galanter, eds., *Handbook of Mathematical Psychology*, vol. II, pages 323–428. New York: John Wiley and Sons.

Chomsky, Noam and Marcel-Paul Schützenberger. 1963. The algebraic theory of context-free languages. In P. Brafford and D. Hirschberg, eds., *Computer Programming and Formal Systems*, pages 118–161. Amsterdam: North-Holland.

Ginsburg, Seymour, Jonathan Goldstine, and Sheila A. Greibach. 1975. Uniformely erasable AFL. *Journal of Comp. and System Sciences* 10:237–247.

Ginsburg, Seymour and Michael A. Harrison. 1967. Bracketed context-free languages. *Journal of Computer and System Sciences* 1(1):1–23.

Harrison, Michael A. 1978. *Introduction to Formal Language Theory*. Addison-Wesley: Reading, MA.

Johnson, Mark. 1996. Left corner transforms and finite state approximation. DRAFT of 12th May, 1996. Rank Xerox Research Centre.

Johnson, Mark. 1998. Finite-state approximation of constraint-based grammars using left-corner grammar transforms. In *Proceedings of the 26th Annual Meeting and 17th International Conference on Computational Linguistics*, vol. 1, pages 619–623. Montreal, Quebec, Canada.

Kappes, Martin. 1998. On the generative capacity of bracketed contextual grammars. *Grammar* 1(2):91–101.

Korenjak, A. J. and John E. Hopcroft. 1966. Simple deterministic languages. In *Proceedings of the Seventh Annual IEEE Symposium on Switching and Automata Theory*, pages 36–46.

Koskenniemi, Kimmo. 1983. Two-level morphology: a general computational model for word-form recognition and production. Publications of the Department of General Linguistics 11, University of Helsinki, Finland.

Krauwer, Steven and Louis des Tombe. 1981. Transducers and grammars as theories of language. *Theoretical Linguistics* 8:173–202.

Langendoen, D. Terence and Yedidyah Langsam. 1987. On the design of finite transducers for parsing phrase-structure languages. In A. Manaster-Ramer, ed., *Mathematics of Language*, pages 191–235. Amsterdam: John Benjamins Publishing Company.

McNaughton, Robert. 1967. Parenthesis grammars. *JACM* 14:490–500.

Mohri, Mehryar and Mark-Jan Nederhof. 2001. Regular approximation of context-free grammars through transformation. In J.-C. Junqua and G. van Noord, eds., *Robustness in Language and Speech Technology*, chap. 9, pages 153–163. Kluwer Academic Publishers.

Mohri, Mehryar and Fernando C. N. Pereira. 1998. Dynamic compilation of weighted context-free grammars. In *36th Annual Meeting of the ACL and 17th International Conference on Computational Linguistics*, vol. 2, pages 891–897. Montreal, Quebec, Canada.

Nederhof, Mark-Jan. 1997. Regular approximation of cfls: a grammatical view. In H. Bunt and A. Nijholt, eds., *International Workshop on Parsing Technologies*, pages 159–170. Massachusetts Institute of Technology.

Nederhof, Mark-Jan. 2000. Practical experiments with regular approximation of context-free languages. *Computational Linguistics* 26(1):17–44.

Pereira, Fernando C. N. and Rebecca N. Wright. 1991. Finite-state approximation of phrase-structure grammars. In *The Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 246–255. Berkeley, California.

Pin, Jean-Eric. 2003. Algebraic tools for the concatenation product. *Theoretical Computer Science* 292(1):317–342.

Pullman, Steven G. 1986. Grammars, parsers, and memory limitations. *Language and Cognitive Processes* 1(3):197–225.

Ritchie, Robert W. and Frederick N. Springsteel. 1972. Language recognition by marking automata. *Information and Control* 20:313–330.

Roche, Emmanuel. 1996. Parsing with finite-state transducers. Tech. Rep. TR-96-30, MERL - A Mitsubishi Electronic Research Laboratory.

Salomaa, Arto. 1973. *Formal Languages*. Academic Press: New York.

Takahashi, Masako. 1975. Generalizations of regular sets and their application to a study of context-free languages. *Information and Control* 27:1–35.

Tapanainen, Pasi. 1997. Applying a finite-state intersection grammar. In E. Roche and Y. Schabes, eds., *Finite-state language processing*, pages 311–327. A Bradford Book, MIT Press, Cambridge, Massachusetts.

Thatcher, James W. 1967. Characterizing derivation trees of context-free grammars through a generalization of finite automata theory. *Journal of Computer and System Sciences* 1(?):317–322.

Wrathall, Celia. 1977. Characterizations of the Dyck sets. *R.A.I.R.O. Informatique théorique/Theoretical Computer Science* 11(1):53–62.

Wu, Dekai. 1995. An algorithm for simultaneously bracketing parallel texts by aligning words. In *ACL-95*, pages 244–251.

Yli-Jyrä, Anssi. 2003a. Describing syntax with star-free regular expressions. In *EACL'2003*. Budapest.

Yli-Jyrä, Anssi. 2003b. Multiplanarity – a model for dependency structures in treebanks. In *Treebanks and Linguistic Theories (TLT 2003)*.